**EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES**

**A PROJECT REPORT**

***Submitted by***

**ANTHU VARSHA. R (921519104001)**

**SNEHA.M (921519104015)**

**TAMIL SELVI.M (921519104017)**

**VEERALAKSHMI.M (921519104019)**

**TEAM ID :PNT2022TMID48918**

**BACHELOR OF ENGINEERING *IN***

**COMPUTER SCIENCE AND**

**ENGINEERING**



|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **TABLE OF CONTENT** | |  |
| **1** | **INTRODUCTION** | |  |
|  | 1.1 | Project Overview |  |
|  | 1.2 | Purpose |  |
| **2** | **LITERATURE SURVEY** | |  |
|  | 2.1 | Existing problem |  |
|  | 2.2 | Reference |  |
|  | 2.3 | Problem Statement Definition |  |
| **3** | **IDEATION & PROPOSED SOLUTION** | |  |
|  | 3.1 | Empathy Map Canvas |  |
|  | 3.2 | Ideation &Brainstorming |  |
|  | 3.3 | Proposed Solution |  |
|  | 3.4 | Problem Solution Fit |  |
| **4** | **REQUIREMENT ANALYSIS** | |  |
|  | 4.1 | Functional requirements |  |
|  | 4.2 | Non-Functional requirements |  |
| **5** | **PROJECT DESIGN** | |  |
|  | 5.1 | Data Flow Diagrams |  |
|  | 5.2 | Solution & Technical Architecture |  |
|  | 5.3 | User Stories |  |
| **6** | **PROJECT PLANNING & SCHEDULING** | |  |
|  | 6.1 | Sprint Planning & Estimation |  |
|  | 6.2 | Sprint Delivery Schedule |  |
|  | 6.3 | Reports from JIRA |  |
| **7** | **CODING & SOLUTIONING (Explain the features**   **added in the project along with code)** | |  |
|  | 7.1 | Feature 1 |  |
|  | 7.2 | Feature 2 |  |
|  | 7.3 | Database Scheme (if applicable) |  |
| **8** | **TESTING** | |  |
|  | 8.1 | Test Cases |  |
|  | 8.2 | User Acceptance Testing |  |
| **9** | **RESULT** | |  |
|  | 9.1 | Performance Metrics |  |
| **10** | **ADVANTAGES &DISADVANTAGES** | |  |
| **11** | **CONCLUSION** | |  |
| **12** | **FUTURE SCOPE** | |  |
| **13** | **APPENDIX** | |  |
|  |  | Source Code |  |
|  |  | GitHub & Project Demo Link |  |

**1.INTRODUCTION**

**1.1.PROJECT OVERVIEW**

It is difficult to predict and detect forest firesin sparsely populated forest areas and it is more difficult if the prediction is done using ground-based models like cameras. Satellites can be an important source of data prior to and also during the fire due to their reliability and efficiency.The various real time forest fire detection and prediction approaches, with the goal of informing the local fire authorities.

**1.2.PURPOSE:**

To detect the forest fire in the early stage.For the early detection of forest fire, we proposed an image recognition system method based on Deep learning model.

**2.LITERATURE SURVEY**

**2.1.EXISTING METHOD:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **AUTHOR** | **TITLE** | **NAME OF JOURNAL** |
| 1. | Medi RahuL, Karnekanti Shiva, SakethAttiliSanjeet and Nenavath Srinivas Naik. | Early Detection of Forest fire using Deep Learning. | 2020.IEEE REGION10 Conference(TENCON),2020,pp.  11361140,doi:10.1109/tencon  50793.2020.9293722. |

1. The system involves pre-processing the image data and applying data augmentation such as shearing, flipping, etc.
2. It uses models like VGG16 , ResNet50 , and DenseNet121 for the classification of images.
3. The model initially divides the train and test sets in 80% and 20% and then sent to the pre-processing phase, where finally it is trained to classify them into two classes fire and non-fire.
4. By using the optimal learning rate the proposed model was able to achieve a training set accuracy of 92.7% and an est set accuracy of 82.57%.

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **AUTHOR** | **TITLE** | **NAME OF JOURNAL** |
| 2. | Byron Arteaga, Mauricio Diaz, Mario jajoa, University of Naino Pasto Columbia . | Deep Learning  Applied forest  Fire Detection. | 2020 IEEE International Symposium on signal processing and information Technology(ISSPIT),2020,pp,  16,doi:10,1109/ISSPIT51521.2020.9408859. |

1. The data processing was done through open source programming language Python, the cloud service Googlecollab, and deep learning algorithms using Pytorch's library.
2. After the data augmentation and pre-processingof the training image, three types of transformation takes place cropping of the image, rotating of an image, and normalizing of the image.
3. The classification of images is done by using the pre-trained models of ResNet and VGG pre-trained models.
4. To validate the performance of each pre-trained model the k-fold method is used.
5. The model obtained during the validationis sent to Raspberry to test its functionality.

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **AUTHOR** | **TITLE** | **NAME OF JOURNAL** |
| 3. | Raghad k. Mohammed(Department of Basic sciences,college of Density, University Baghdad,Baghdad,Iraq). | A Real-time forest fire and  Smoke detection  System Using  Deep Learning. | International Journal of Nonlinear Analysis and Application 13.1(2022):2053-  2063. |

1. The proposed framework aims to detect smoke and fire based on the images received from the video stream from the Raspberry Pi
2. Pre-processing of image data.
3. Image data augmentation (Scale, horizontal flip, and vertical flip).
4. Pre-trainingng model imagenet dataset ->{inception-ResNet-V2}.
5. By fine-tuning the above two steps we have to send that to the fully connected layer with softmax.

* we can view the model accuracy as instead.

**2.2.REFERENCES:**

1.Early detection of forest fire - <https://ieeexplore.ieee.org/document/9293722>

using deep learning.

2.Deep Learning Applied -<https://ieeexplore.ieee.org/document/9408859>

Forest fire Detection.

3.A Real-time Forest Fire Smoke detection - <https://ijnaa.semnan.ac.ir/article_5899.html>

System Using Deep Learning.

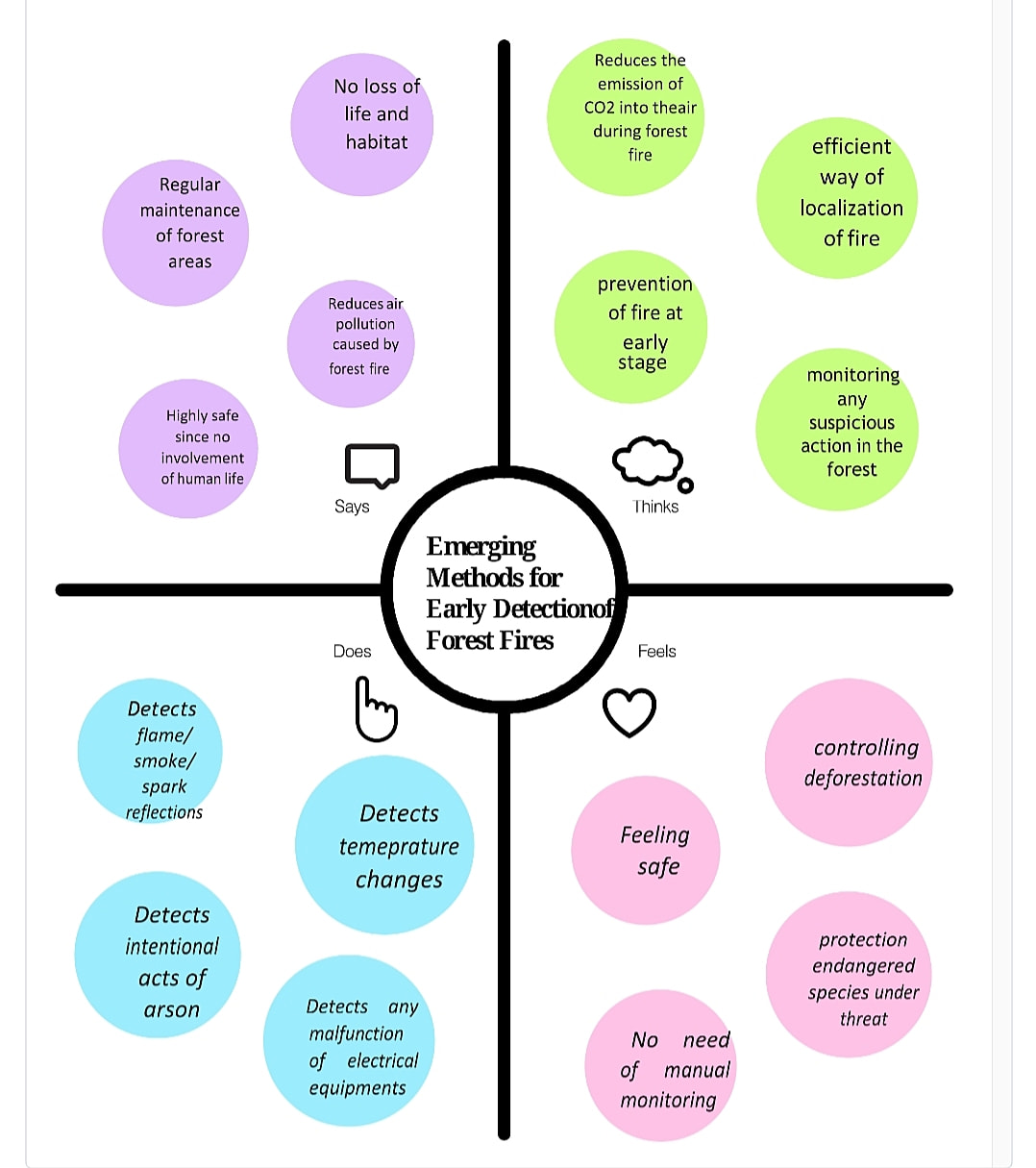
**2.3.PROBLEM STATEMENT DEFINITION**

Forest fires is a wide spread and critical factor in the earth’s ecosystem. The most effective and vital solution is early detection fires to preserve natural resources and to protect living creatures.

|  |  |
| --- | --- |
| Who does the problem affect? | People living in the forest. |
| When does the issue occurs? | When there is a climate change in the environment . |
| Where is the issue occurring? | The issue occurs when there is a difficulty to identify the forest fires. |
| What is the issue? | Forest fires are a major environmental issue,creating economic and ecological damage while endangering human lives. |
| Why is it important that we fix the problem? | By solving these issues,it can reduce the forest fire in the beginning stage,by alerting user and can save the ecosystem and human lives. |

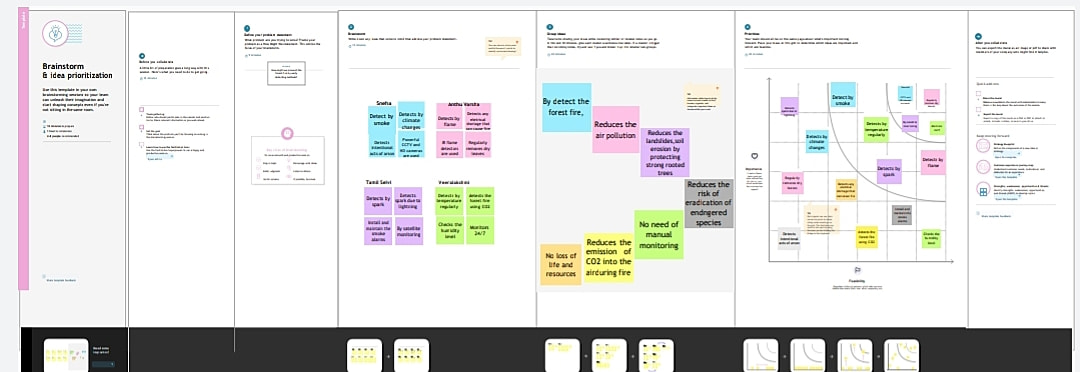
**3.IDEATION & PROPOSED SOLUTION**

**3.1.**EMPATHY MAP CANVAS

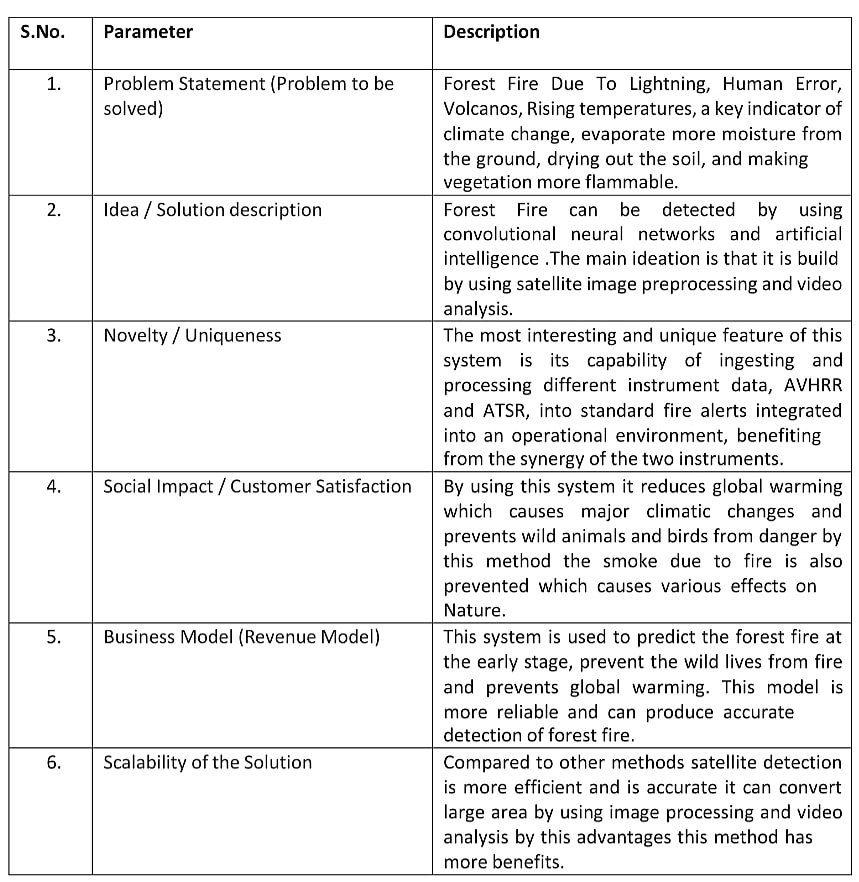


**3.2.BRAINSTORMING :**

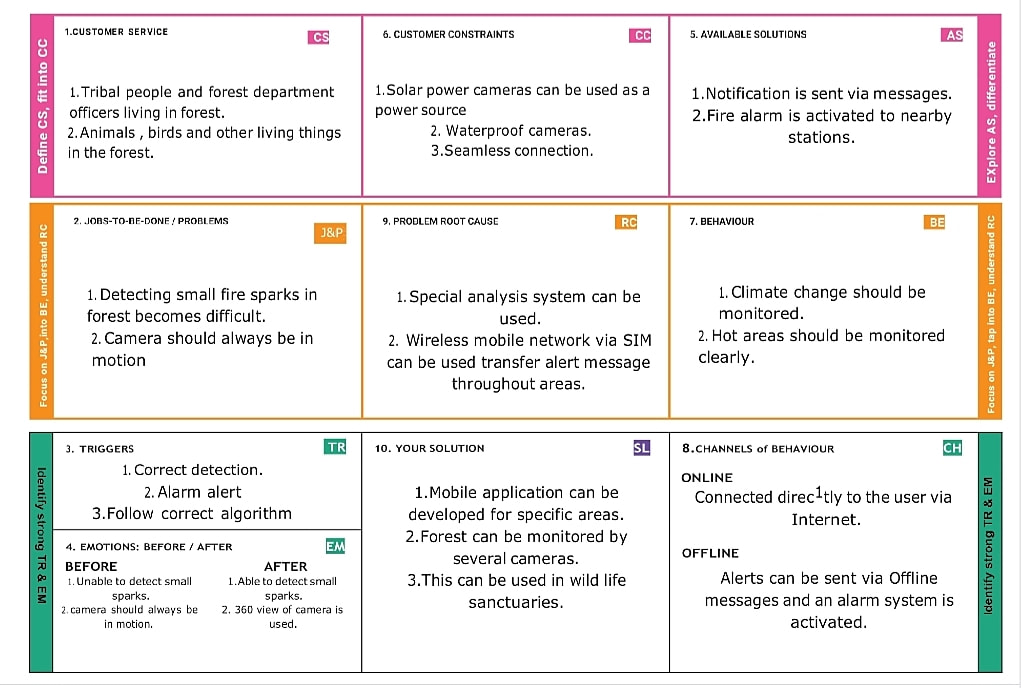
**Problem Statements:**



**3.3.PROPOSED SOLUTION:**

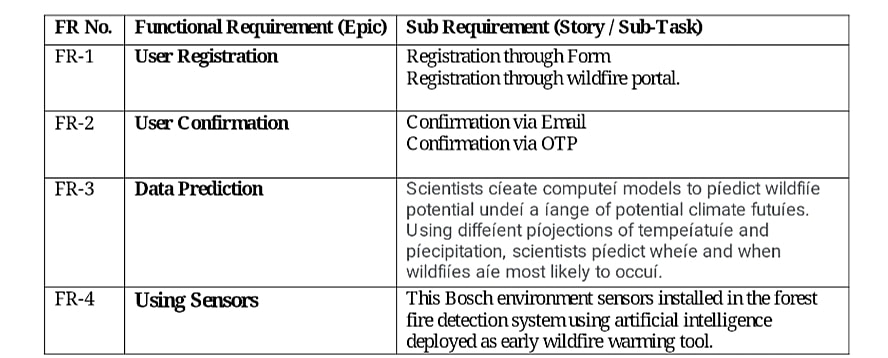


**3.4.PROPOSED SOLUTION FIT:**

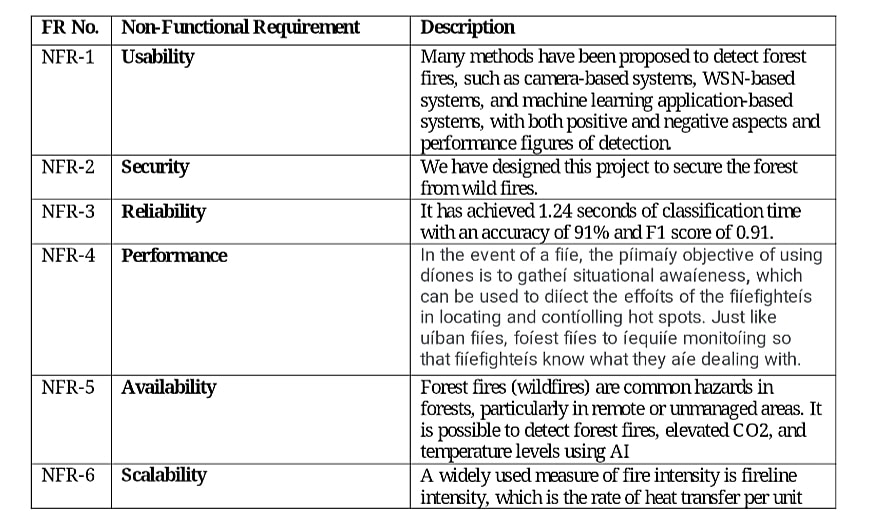


**4.REQUIREMENT ANALYSIS**

**4.1.FUNCTIONAL REQUIREMENTS:**

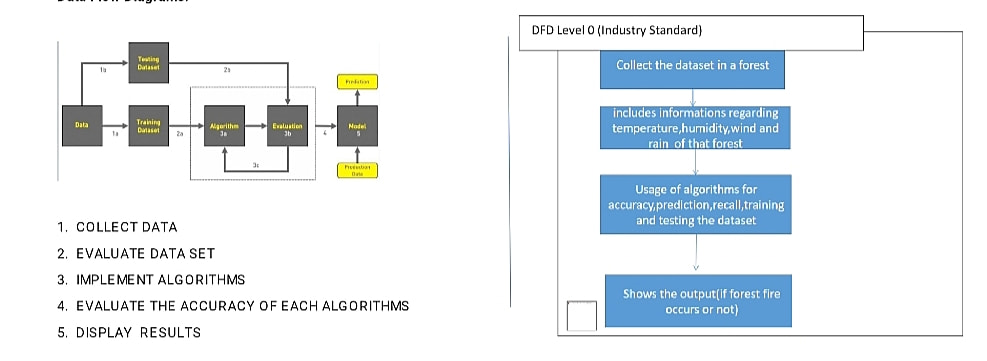


**4.2.**NON-FUNCTIONAL REQUIREMENTS

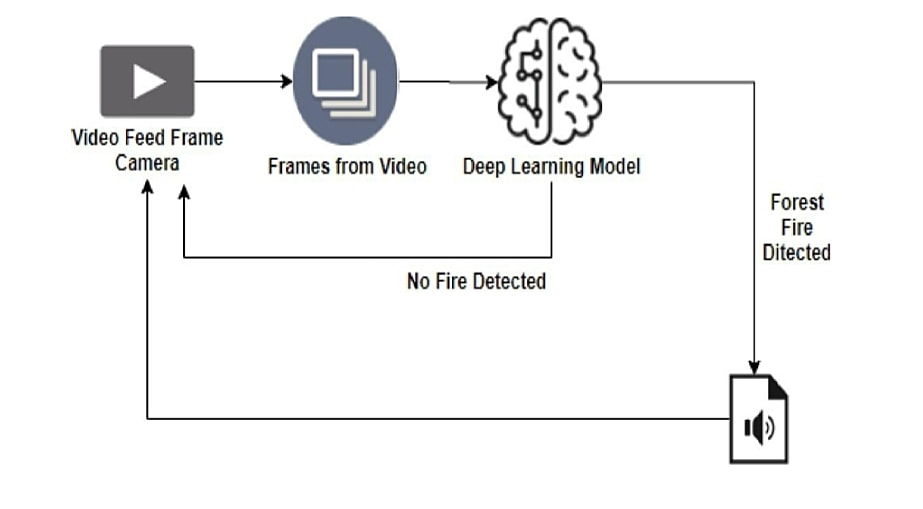


**5.PROJECT DESIGN**

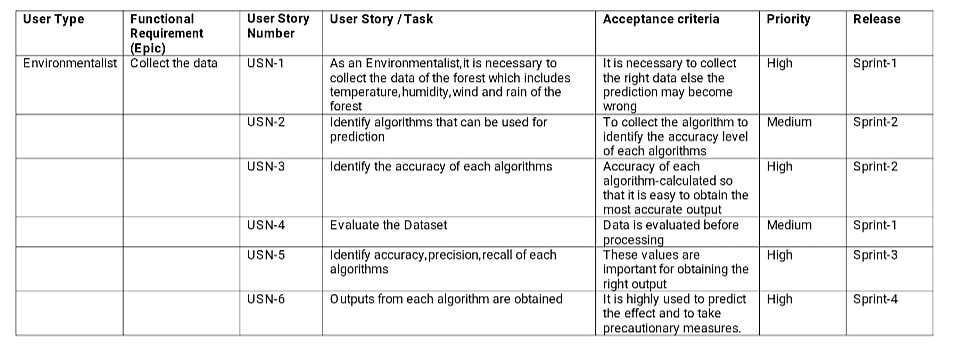
**5.1.DATA FLOW DIAGRAMS:**



**5.2.SOLUTION AND TECHNICAL ARCHIETECTURE:**

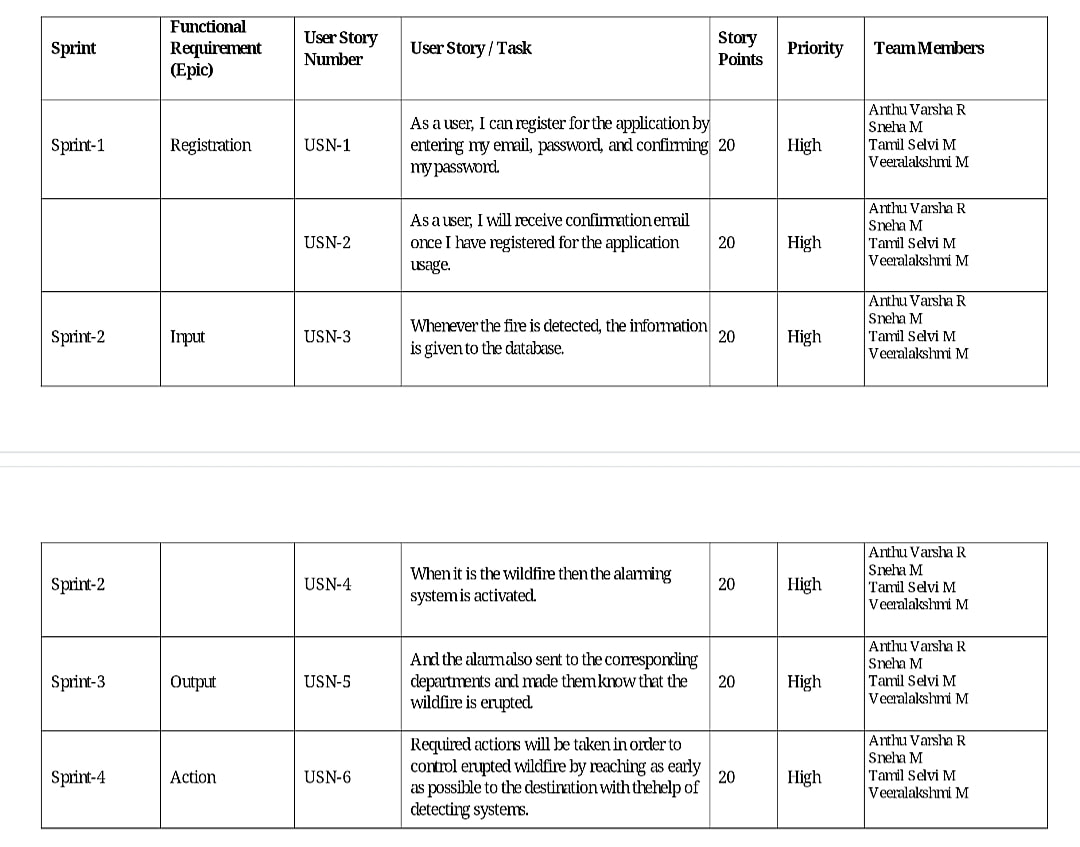


**5.3.USER STORIES:**

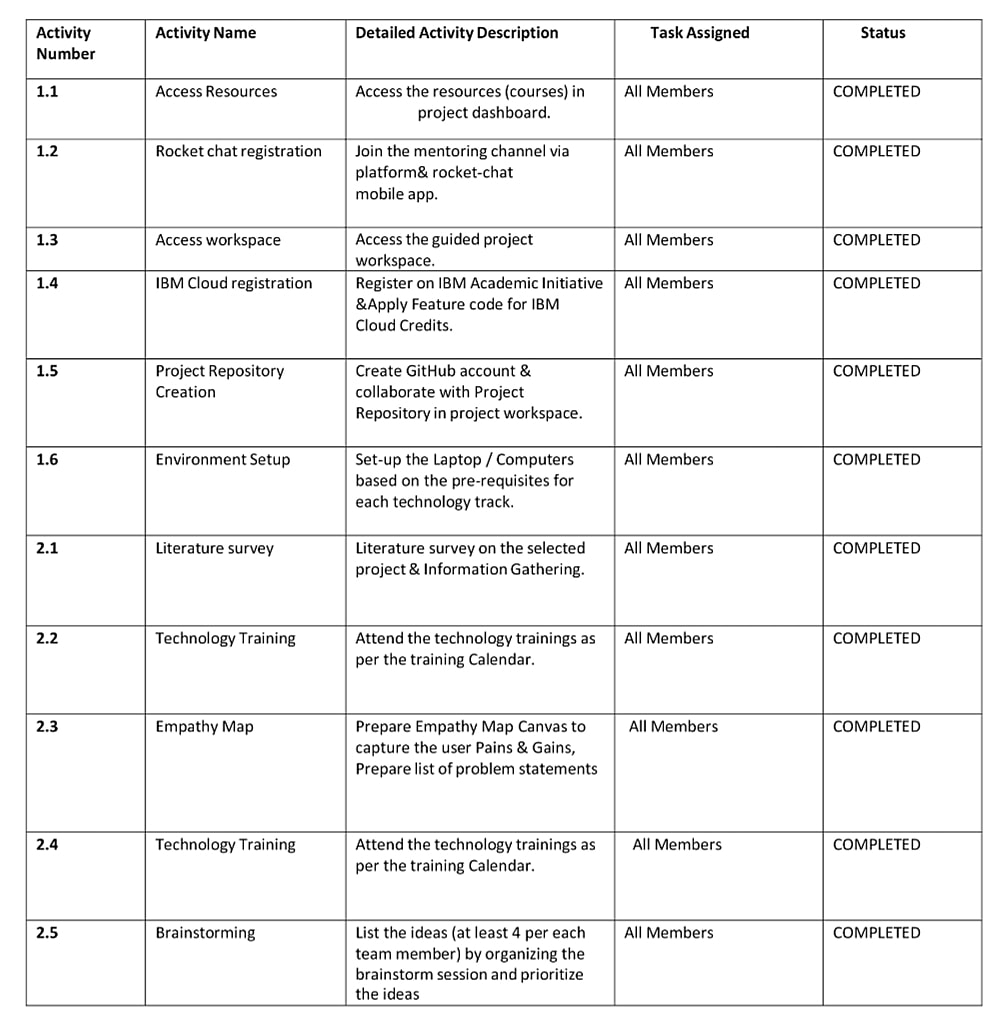


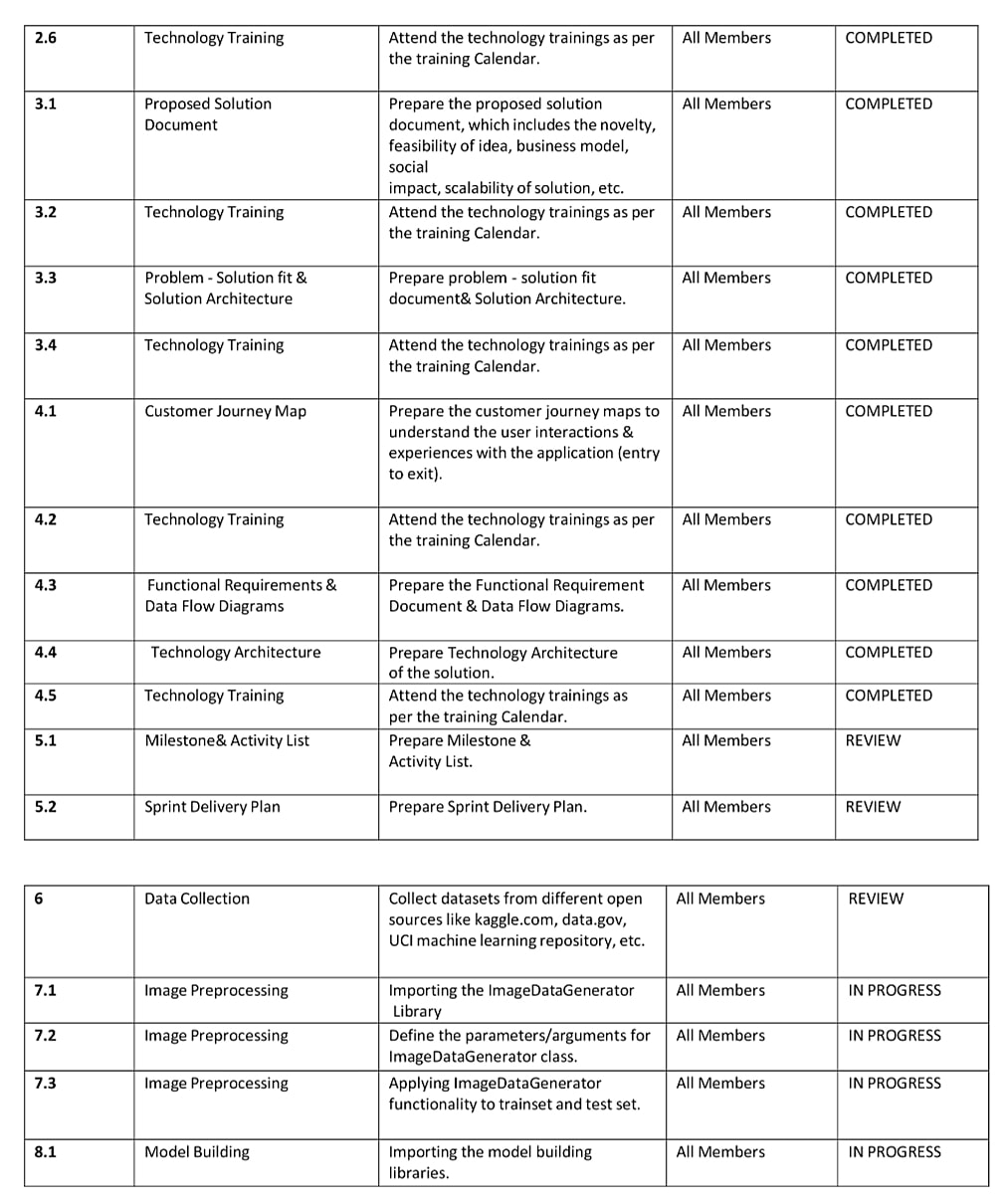
**6.PROJECT PLANNING & SCHEDULING:**

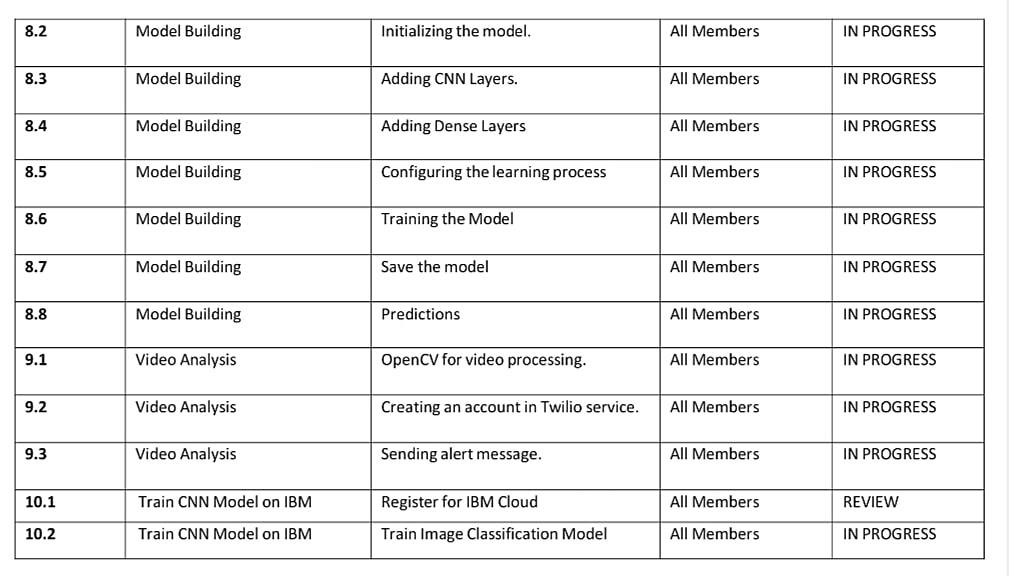
**6.1.SPRINT PLANNING & ESTIMATION:**



MILESTONE & ACTIVITY LIST:

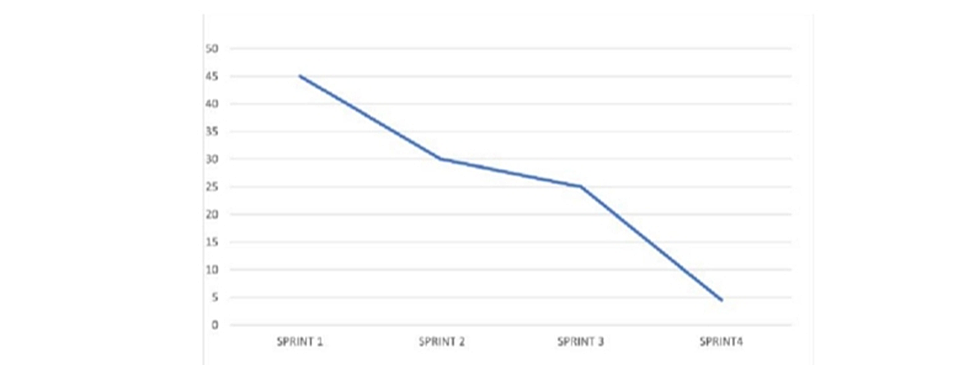






**6.2.SPRINT DELIVERY SCHEDULE:**

**6.3.REPORTS FROM JIRA:**



**7.CODING & SOLUTIONING**

**7.1.FEATURE 1:**

**1.IMAGEDATA GENERATOR**

Keras ImageDataGenerator is used for getting the input of the original data and further, it makes the transformation of this data on a random basis and gives the output resultant containing only the data that is newly transformed. It does not add the data.

from keras.preprocessing.image import ImageDataGenerator

**2.PARAMETRES**

**2.1.Rescale:**

The ImageDataGenerator class can be used to rescale pixel values from the range of 0-255 to the range 0-1 preferred for neural network models. Scaling data to the range of 0-1 is traditionally referred to as normalization.

**2.2.Shear Range:**

Shear range means that the image will be distorted along an axis, mostly to create or rectify the perception angles. It's usually used to augment images so that computers can see how humans see things from different angles.

**2.3.Rotation range:**

ImageDataGenerator class allows you to randomly rotate images through any degree between 0 and 360 by providing an integer value in the rotation\_range argument. When the image is rotated, some pixels will move outside the image and leave an empty area that needs to be filled in.

**2.4.Zoom Range:**

The zoom augmentation method is used to zooming the image. This method randomly zooms the image either by zooming in or it adds some pixels aroundthe image to enlarge the image. This method uses the zoom\_range argument of the ImageDataGenerator class. We can specify the percentage value of the zooms either in a float**,** range in the form of an array.

**2.5.Horizontal Flip:**

Horizontal flip basically flips both rows and columns horizontally. So for this, we have to pass the horizontal\_flip=True argument in the ImageDataGenerator constructor.

**3.CONVOLUTION NEURAL NETWORK:**

A CNN is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data. There are other types of neural networks in deep learning, but for identifying and recognizing objects, CNNs are the network architecture of choice. The layers used in the CNN algorithm is Convolutional ,maxpooling, and flatten layer.

**3.1.Convolutional Layer:**

A convolutional layer is the main building block of a CNN. It contains a set of filters (or kernels), parameters of which are to be learned throughout the training. The size of the filters is usually smaller than the actual image. Each filter convolves with the image

Convolution layer is used for a image processing to blur and sharpen images, but also to perform other operations.

from keras.layers import Convolution2D

**3.2.Maxpooling Layer:**

Max pooling is a pooling operation that selects the maximum element from the region of the feature map covered by the filter.

from keras.layers import MaxPooling2D

**3.3.Flatten Layer:**

Flattening is used to convert all the resultant 2-Dimensional arrays from pooled feature maps into a single long continuous linear vector. The flattened matrix is fed as input to the fully connected layer to classify the image.

from keras.layers import Flatten

**4.DENSE LAYER:**

Dense Layer is used to classify image based on output from convolutional layers.

**7.2.FEATURE 2(CODE):**

**Importing Keras libraries**

import keras

**Importing ImageDataGenerator from Keras**

from matplotlib import pyplot as plt

from keras.preprocessing.image import ImageDataGenerator

**Defining the Parameters**

train\_datagen=ImageDataGenerator(rescale=1./255,shear\_range=0.2,rotation\_range=180,zoom\_range=0.2,horizontal\_flip=True)

test\_datagen=ImageDataGenerator(rescale=1./255,shear\_range=0.2,rotation\_range=180,zoom\_range=0.2,horizontal\_flip=True)

**Applying ImageDataGenerator functionality to train dataset**

from google.colab import drive

drive.mount('/content/drive')

x\_train**=**train\_datagen**.**flow\_from\_directory('/content/drive/MyDrive/IBM/Dataset/Dataset/train\_set',

target\_size**=**(128,128),batch\_size**=**32,class\_mode**=**'binary')

**Applying ImageDataGenerator functionality to test dataset**

x\_test**=**test\_datagen**.**flow\_from\_directory('/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set',

target\_size**=**(128,128),batch\_size**=**32,class\_mode**=**'binary')

**Importing Model Building Libraries**

#to define the linear Initialisation import sequential

from keras.models import Sequential

#to add layers import Dense

from keras.layers import Dense

#to create Convolutional kernel import convolution2D

from keras.layers import Convolution2D

#import Maxpooling layer

from keras.layers import MaxPooling2D

#import flatten layer

from keras.layers import Flatten

import warnings

warnings.filterwarnings('ignore')

**Initializing the model**

model = Sequential()

**Adding CNN Layers**

model.add(Convolution2D(32,(3,3),input\_shape=(64,64,3),activation='relu'))

#add maxpooling layers

model.add(MaxPooling2D(pool\_size=(2,2)))

#add faltten layer

model.add(Flatten())

**Add Dense layers**

#add hidden layers

model.add(Dense(150,activation='relu'))

#add output layer

model.add(Dense(1,activation='sigmoid'))

**configuring the learning process**

model.compile(loss='binary\_crossentropy',optimizer="adam",metrics=["accuracy"])

**Training the model**

model.fit\_generator(x\_train,steps\_per\_epoch=14,epochs=10,validation\_data=x\_test,validation\_steps=4)

**Save the model**

model.save("forest.h5")

**Predictions**

#import load model from keras.model

from keras.models import load\_model

#import image from keras

from tensorflow.keras.preprocessing import image

import numpy as np

#import cv2

import cv2

#load the saved model

model=load\_model("forest.h5")

img=image.load\_img('/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/forest/0.64133000\_1519374442\_forest\_deep.jpg')

x=image.img\_to\_array(img)

res=cv2.resize(x,dsize=(64,64),interpolation=cv2.INTER\_CUBIC)

#expand the image shape

x=np.expand\_dims(res,axis=0)

pred=model.predict(x)

pred = int(pred[0][0])

pred

int(pred)

pip install twilio

from twilio.rest import Client

**if** pred**==**0:

print('Forest fire')

account\_sid**=**'AC2a45c94082d15fd26a52ef4274bc95bd'

auth\_token**=**'769fa5108da21ef101aba8ef9ee8ce73'

client**=**Client(account\_sid,auth\_token)

message**=**client**.**messages \

**.**create(

body**=**'forest fire is detected,stay alert',

*#use twilio free number*

from\_**=**'+18316035766',

*#to number*

to**=**'+916383605324')

print(message**.**sid)

print("No Danger")

print("SMS Sent!")

**elif** pred**==**1:

print('Fire Detected')

**Open cv for video processing**

pip install twilio

from logging import WARNING

#import opencv library

import cv2

#import numpy

import numpy as np

#import image function from keras

from keras.preprocessing import image

#import load\_model from keras

from keras.models import load\_model

#import client from twilio API

from twilio.rest import Client

#import playsound package

**Creating An Account in Twilio Service**

**Sending Alert Message**

**import** cv2

**import** numpy **as** np

**from** google.colab.patches **import** cv2\_imshow

**from** matplotlib **import** pyplot **as** plt

**import** librosa

**from** tensorflow.keras.preprocessing **import** image

**from** keras.models **import** load\_model

*# Create a VideoCapture object and read from input file*

*# If the input is the camera, pass 0 instead of the video file name*

cap **=** cv2**.**VideoCapture('/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/with fire/19464620\_401.jpg')

*# Check if camera opened successfully*

**if** (cap**.**isOpened()**==** **False**):

print("Error opening video stream or file")

*# Read until video is completed*

**while**(cap**.**isOpened()):

*# Capture frame-by-frame*

ret, frame **=** cap**.**read()

**if** ret **==** **True**:

x**=**image**.**img\_to\_array(frame)

res**=**cv2**.**resize(x,dsize**=**(128,128),interpolation**=**cv2**.**INTER\_CUBIC)

*#expand the image shape*

x**=**np**.**expand\_dims(res,axis**=**0)

model**=**load\_model("/content/drive/MyDrive/IBM/forest1.h5")

cv2\_imshow(frame)

pred**=**model**.**predict(x)

pred **=** int(pred[0][0])

pred

int(pred)

**if** pred**==**0:

print('No danger')

**break**

**else**:

print("Forest fire")

**break**

*# When everything done, release the video capture object*

cap**.**release()

*# Closes all the frames*

cv2**.**destroyAllWindows()

**from** twilio.rest **import** Client

**from** playsound **import** playsound

**if** pred**==**0:

print('Forest fire')

account\_sid**=**'AC2a45c94082d15fd26a52ef4274bc95bd'

auth\_token**=**'769fa5108da21ef101aba8ef9ee8ce73'

client**=**Client(account\_sid,auth\_token)

message**=**client**.**messages \

**.**create(

body**=**'forest fire is detected,stay alert',

*#use twilio free number*

from\_**=**'+18316035766',

*#to number*

to**=**'+916383605324')

print(message**.**sid)

print("No Danger")

print("SMS Sent!")

**elif** pred**==**1:

print('Fire Detected'

**8.TESTING**

**8.1.Test Cases:**

**8.2.User Acceptance Testing:**

**Purpose of Document:**

The purpose of this document is to briefly explain the test coverage and open issues of the [Early detection of forest fire using Deep Learning] project at the time of the release to User Acceptance Testing (UAT).

**Defect Analysis:**

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Resolution** | **Severity1** | **Severity2** | **Severity3** | **Severity4** | **Subtotal** |
| By Design | 5 | 1 | 1 | 1 | 8 |
| Duplicate | 1 | 0 | 3 | 0 | 4 |
| External | 2 | 3 | 0 | 1 | 6 |
| Fixed | 7 | 2 | 4 | 10 | 23 |
| Not Reproduced | 0 | 0 | 0 | 0 | 0 |
| Skipped | 0 | 0 | 1 | 1 | 2 |
| Won'tFix | 0 | 3 | 2 | 1 | 6 |
| Totals | 15 | 9 | 11 | 14 | 49 |

**Test Case Analysis:**

This report shows the number of test cases that have passed, failed, and untested

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Section** | **Total Cases** | **Not Tested** | **Fail** | **Pass** |
| Print Engine | 5 | 0 | 0 | 5 |
| Client Application | 30 | 0 | 0 | 30 |
| Security | 2 | 0 | 0 | 2 |
| Out source Shipping | 3 | 0 | 0 | 3 |
| Exception Reporting | 9 | 0 | 0 | 9 |
| Final Report Output | 4 | 0 | 0 | 4 |
| Version Control | 2 | 0 | 0 | 2 |

**9.RESULTS**

**9.1.PERFORMANCSE METRICS:**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Values** |
|  | Model Summary | As a threat of forest fire increases due to climate changes, the need for finding a detection system increase .We proposed a Deep Learning-based model for early detection of forest fire. The Proposed model successfully classifies the images into fire and no fire, and sends an alert messages in case of fire. Thus, the Deep Learning algorithms proved their efficiency in detecting different objects. |
|  | Accuracy | Training Accuracy - 92% - 98%   Validation Accuracy - 95% |

**10.ADVANTAGES & DISADVANTAGES**

**ADVANTAGES:**

1.Ability to cover areas at different altitudes and locations.

2.The results is quite accurate with the accuracy upto 92%

3.Reliability - The model is very effective, inexpensive and easy to apply.

4.The model, it shows the 'fire' and 'no fire' images classified with high accuracy.

5.Video analysis of this model leads to low degree of misjudgment of fire detection.

**DISADVANTAGES:**

1.Individual learner is responsible for learning global information to avoid false positives.

2.The limited learning and perception ability of individual learners is not sufficient to make them perform well in complex tasks.

3.Proper connectivity and maintenance will be a complex task.

**11.CONCLUSION**

As a threat of forest fire increases due to climate changes, the need for finding a detection system increase .We proposed a Deep Learning-based model for early detection of forest fire. The Proposed model successfully classifies the images into fire and no fire, and sends an alert messages in case of fire. Thus, the Deep Learning algorithms proved their efficiency in detecting different objects.

**12.FUTURE SCOPE**

1. Integrate live satellite data and process real time processing of the fires.
2. Enchance the time complexity of the detection of forest fires to improve the speed.
3. These accidents can be controlled to a greater extend.
4. Forest fire leads to destruction of excess of species, by using this technique we can save the life and environment

**13.APPENDIX**

**SOURCE CODE:**

Our Github link- [https://github.com/IBM-EPBL/IBM-Project-523621660998368.git](https://github.com/IBM-EPBL/IBM-Project-52362-1660998368.git)

**Y**outube link:

<https://youtu.be/eJAFY_VpxeI>